Programare Logică – Temă Colectivă de Laborator*

Claudia MUREŞAN
Universitatea din Bucureşti, Facultatea de Matematică şi Informatică
c.muresan@yahoo.com, cmuresan@fmi.unibuc.ro, claudia.muresan@unibuc.ro

2020–2021, Semestrul II

Exerciţiul 1. După modelul sortării prin inserţie directă după un criteriu arbitrar, adică după o relaţie de ordine (fără eliminarea duplicatelor) sau ordine strictă (cu eliminarea duplicatelor) arbitrară efectuate la laborator, să se modifice sortarea prin metoda bulelor, sortarea prin interclasare binară şi sortarea rapidă efectuate la laborator astfel încât să efectueze sortarea unei liste după un criteriu arbitrar.

Exerciţiul 2. Să se scrie în Prolog două predicate binare **lex** şi **lexterm**, definite astfel:
• $lex(L, M)$ (sau, dacă e notat infixat, $L\ lex\ M$) e satisfăcut ddacă L şi M sunt liste de numere, iar L este mai mică sau egală decât M în ordine lexicografică, adică, dacă $L = [N_1, N_2, \ldots, N_n]$ pentru un $n \in \mathbb{N}$, iar $L = [P_1, P_2, \ldots, P_p]$ pentru un $p \in \mathbb{N}$, atunci există $k \in \mathbb{N}$ astfel încât:
$k \leq n$ şi $k \leq p$,
pentru fiecare $i \in \overline{1, k}$, $N_i = P_i$,
şi fie $k = n$, fie: $k < n$ şi $k < p$ şi $N_{k+1} < P_{k+1}$;
• $lexterm(L, M)$ (sau, dacă e notat infixat, $L\ lexterm\ M$) e satisfăcut ddacă L şi M sunt liste de termeni Prolog, iar L este mai mică sau egală decât M în ordine lexicografică în raport cu ordinea nestrictă pe termeni, adică, dacă $L = [N_1, N_2, \ldots, N_n]$ pentru un $n \in \mathbb{N}$, iar $L = [P_1, P_2, \ldots, P_p]$ pentru un $p \in \mathbb{N}$, atunci există $k \in \mathbb{N}$ astfel încât:
$k \leq n$ şi $k \leq p$,
pentru fiecare $i \in \overline{1, k}$, $N_i = P_i$,
şi fie $k = n$, fie: $k < n$ şi $k < p$ şi $N_{k+1} @< P_{k+1}$.

Exerciţiul 3. Să se scrie în Prolog trei predicate binare **sumar**, **maxar** şi **maxsubterm**, definite astfel:
• $sumar(Termen, SumaAritati)$ e satisfăcut ddacă $Termen$ e un termen Prolog, iar $SumaAritati$ este suma arităţilor operatorilor din componenţa lui $Termen$, definită recursiv astfel:
dacă $Termen$ e o variabilă sau o constantă, atunci $SumaAritati = 0$;
dacă $Termen = f(T_1, \ldots, T_n)$, unde $n \in \mathbb{N}^*$, f este un operator n–ar, iar $T_1, \ldots, T_n$ sunt termeni Prolog, atunci $SumaAritati = n + SumAr_1 + \ldots + SumAr_n$, unde $sumar(T_1, SumAr_1), \ldots, sumar(T_n, SumAr_n)$ sunt satisfăcute;
iar, într–o interogare în Prolog, pentru un termen dat ca prim argument, *sumar* să calculeze, ca mai sus, suma arităţilor operatorilor dominanţi ai subtermenilor săi în al doilea argument;
• $maxar(Termen, AritateMaxima)$ e satisfăcut ddacă $Termen$ e un termen Prolog, iar $AritateMaxima$ este aritatea maximă a unui operator din componenţa lui $Termen$, definită recursiv astfel:
dacă $Termen$ e o variabilă sau o constantă, atunci $AritateMaxima = 0$;
dacă $Termen = f(T_1, \ldots, T_n)$, unde $n \in \mathbb{N}^*$, f este un operator n–ar, iar $T_1, \ldots, T_n$ sunt termeni Prolog, atunci $AritateMaxima = \max\{n, ArMax_1, \ldots, ArMax_n\}$, unde $maxar(T_1, ArMax_1), \ldots, maxar(T_n, ArMax_n)$ sunt satisfăcute (iar maximul este, desigur, raportat la ordinea naturală =< pe numere);
iar, într–o interogare în Prolog, pentru un termen dat ca prim argument, *maxar* să calculeze, ca mai sus, maximul arităţilor operatorilor dominanţi ai subtermenilor săi în al doilea argument;
• $maxsubterm(Termen, SubtermenMaxim)$ e satisfăcut ddacă $Termen$ e un termen Prolog, iar $SubtermenMaxim$ este subtermenul lui $Termen$ maxim în raport cu ordinea @ =< pe termeni Prolog, definit astfel:
dacă $Termen$ e o variabilă sau o constantă, atunci $SubtermenMaxim = Termen$;
dacă $Termen = f(T_1, \ldots, T_n)$, unde $n \in \mathbb{N}^*$, f este un operator n–ar, iar $T_1, \ldots, T_n$ sunt termeni Prolog, atunci $SubtermenMaxim = \max(\{Termen, SubtermMax_1, \ldots, SubtermMax_n\}, @ =<)$, unde $maxsubterm(T_1, SubtermMax_1), \ldots, maxsubterm(T_n, SubtermMax_n)$ sunt satisfăcute;
iar, într–o interogare în Prolog, pentru un termen dat ca prim argument, *maxsubterm* să calculeze, ca mai sus, subtermenul său maxim în al doilea argument.

*Obligatorie pentru seria 31: de trimis într–un singur exemplar de fiecare grupă/semigrupă şi prezentat la laborator. Facultativă pentru seria ID: de trimis într–un singur exemplar de întreaga serie.

Exerciţiul 4. Folosind câte una dintre sortările implementate la Exerciţiul 1, să se scrie predicate binare pentru:

① sortarea unei liste de liste de numere după ordinea lexicografică *lex* implementată la Exerciţiul 2;

② sortarea unei liste de liste de termeni după ordinea lexicografică *lexterm* implementată la Exerciţiul 2;

③ sortarea unei liste de termeni crescător nestrict după sumele arităţilor operatorilor din componenţa acelor termeni, calculate cu predicatul *sumar* implementat la Exerciţiul 3;

④ sortarea unei liste de termeni crescător nestrict după maximele arităţilor operatorilor din componenţa acelor termeni, calculate cu predicatul *maxar* implementat la Exerciţiul 3;

⑤ sortarea unei liste de termeni după ordinea @ =< pe subtermenii maximi ai acelor termeni, determinaţi cu predicatul *maxsubterm* implementat la Exerciţiul 3.

Exerciţiul 5. Să se scrie în Prolog un predicat binar **reordterm** care, într–o interogare în Prolog, dacă primeşte un termen T ca prim argument, construieşte în al doilea argument termenul U obţinut din T prin reordonarea subtermenilor săi după ordinea @ =<, recursiv, astfel:

dacă T e o variabilă sau o constantă, atunci $U = T$;

dacă $T = f(T_1, \ldots, T_n)$, unde $n \in \mathbb{N}^*$, f este un operator n–ar, iar $T_1, \ldots, T_n$ sunt termeni Prolog, atunci $U = f(U_1, \ldots, U_n)$, unde $[V_1, \ldots, V_n]$ este lista de termeni Prolog obţinută din lista $[T_1, \ldots, T_n]$ prin sortarea lexicografică raportat la ordinea @ =< implementată la Exerciţiul 4, punctul ②, iar $U_1, \ldots, U_n$ sunt termenii care satisfac: $reordterm(V_1, U_1), \ldots, reordterm(V_n, U_n)$.